Similarity Search for Scientific Workflows

Johannes Starlinger* Bryan Brancotte† Sarah Cohen-Boulakia†‡ Ulf Leser*

*Humboldt-Universität zu Berlin
Institut für Informatik
Unter den Linden 6
10099 Berlin, Germany
{starling,leser}@informatik.hu-berlin.de

†Université Paris-Sud
Laboratoire de Recherche en Informatique
CNRS UMR 8623, France
‡And INRIA, LIRMM, IBC, France
{brancotte,cohen}@lri.fr

ABSTRACT

With the increasing popularity of scientific workflows, public repositories are gaining importance as a means to share, find, and reuse such workflows. As the sizes of these repositories grow, methods to compare the scientific workflows stored in them become a necessity, for instance, to allow duplicate detection or similarity search. Scientific workflows are complex objects, and their comparison entails a number of distinct steps from comparing atomic elements to comparison of the workflows as a whole. Various studies have implemented methods for scientific workflow comparison and came up with often contradicting conclusions upon which algorithms work best. Comparing these results is cumbersome, as the original studies mixed different approaches for different steps and used different evaluation data and metrics. We contribute to the field (i) by dissecting each previous approach into an explicitly defined and comparable set of subtasks, (ii) by comparing in isolation different approaches taken at each step of scientific workflow comparison, reporting on an number of unexpected findings, (iii) by investigating how these can best be combined into aggregated measures, and (iv) by making available a gold standard of over 2000 similarity ratings contributed by 15 workflow experts on a corpus of almost 1500 workflows and re-implementations of all methods we evaluated.

1. INTRODUCTION

Over the past decade, scientific workflows have established themselves as a valuable means for scientists to create reproducible in-silico experiments [12]. Several scientific workflow management systems (SWFM) have become freely available, easing scientific workflow creation, management, and execution, such as Taverna, Kepler, VisTrails, Galaxy, Pegasus, Knime, e-BioFlow, e-Science Central, or Askalon (e.g., [30, 6, 16, 20]). Yet, creating scientific workflows using an SWFM is still a laborious task and complex enough to impede non computer-savvy researchers from using these tools [10]. Online repositories have emerged to allow sharing of scientific workflows, facilitating their reuse and repurposing. Popular examples of such scientific workflow repositories include CrowdLabs [28], SHIWA [2], the repositories offered with Kepler [6] and Galaxy [20], or myExperiment [32], currently containing more than 2500 workflows from various disciplines including bioinformatics, astrophysics, earth sciences, or particle physics.

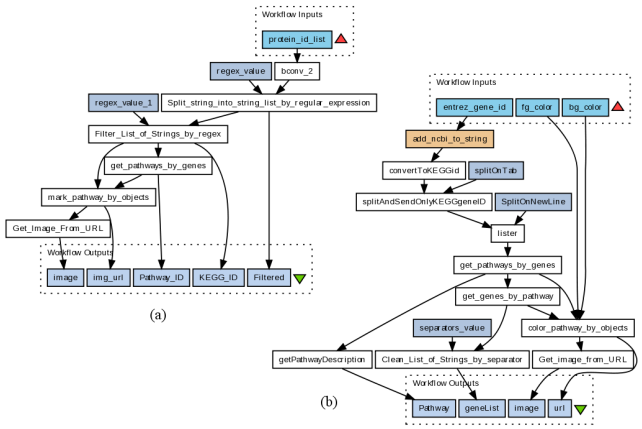


Figure 1: Sample scientific workflows from myExperiment: (a) ID: 1189, Title: *KEGG pathway analysis*, (b) ID: 2805, Title: *Get Pathway-Genes by Entrez gene id*.

Figure 1 shows two exemplary scientific workflows from the myExperiment repository. Scientific workflows typically model a dataflow with a structure resembling a directed acyclic graph (DAG). They have global inputs and outputs, data processing modules which operate on the data, and datalinks which connect the former and thereby define the flow of data from one module to the next. Each module has attributes associated with it, such as a label, input and output signatures, the type of operation to be carried out, and, if applicable, a set of static, data independent parameters, such as the url of a web-service to be invoked. Upon upload to a repository, workflows typically are further annotated with a title, a general description of their functionality, keyword tags, and the uploading author.

With increasing repository sizes, new challenges arise for managing these collections of scientific workflows and for using the information collected in them as a source of expert-supplied knowledge [10, 19]. Challenges include the detection of functionally equivalent workflows, grouping of work-

Proceedings of the VLDB Endowment, Vol. 7, No. 12

flows into functional clusters, workflow retrieval, or the use of existing workflows in the design of novel workflows [36, 34, 33, 4, 18]. The core operation necessary for meeting any of these challenges is the algorithmic comparison of two workflows regarding their functional similarity. Existing approaches to such similarity measures for scientific workflows are usually classified as being either annotation-based or structure-based, depending on which of the information described above they use. Each of these classes of algorithms has its particular strengths and weaknesses: Annotation-based approaches are independent of the workflows' formats and can be used to compare workflows both across different SWFM, and across multiple repositories [11]. Yet, they only work if the workflows under comparison have annotations, which may or may not be the case for a workflow stored in a public repository by an arbitrary user. Approaches for structural workflow comparison, on the other hand, can be applied without such backing human-provided textual knowledge. Yet, they have to assess a workflow's functionality from the information contained in its DAG structure and the modules it is composed of. As another source of data, provenance represents concrete execution traces of workflows. Such traces would allow, for instance, to take execution parameters and runtime information into account as an additional means of comparison. However, specialized provenance databases have just started to emerge (e.g., in the ProvBench initiative [1]), and we are not aware of any workflow repository also containing real execution traces.

Which approach to scientific workflow comparison provides best results, and how different aspects of workflows contribute to their functional similarity is still an open question. There have been numerous studies investigating both annotational [11, 36, 17] and structural [34, 33, 36, 4, 18, 17, 38] approaches, but their comparison is hindered by a number of factors. Firstly, the process of scientific workflow comparison entails several steps from comparison of single modules to comparison of whole workflows [35, 4] - each of which may be treated differently in the methods considered. This makes it hard to determine how single aspects contribute to workflow similarity, and which approach to a specific step of the comparison process provides best results. Secondly, the evaluation of a proposed method is often done by manual inspection of the methods concrete output, or on a proprietary dataset, both hampering repeatability. To compare multiple methods and configurations it is necessary to have a method-independent, gold-standard corpus to evaluate on. To the best of our knowledge, for scientific workflow similarity a gold-standard corpus of decent size does not exist, yet. Reference corpora exist for business process models [14, 15, 13] but these cannot be used easily because (1) they often come without gold standard ratings and (2) business workflows typically contain rich control structures calling for other similarity measures than purely data-driven scientific workflows [25, 39]. Other work uses synthetic workflows to test similarity measures (e.g.,[21]), while we focus on real-life workflows. Thirdly, the presented evaluations vary with the underlying use case. For instance, similarity measures are a requirement for both clustering and similarity search. The results derived from the corresponding evaluations are difficult to compare.

Addressing these issues, we here present results of a comprehensive re-implementation and evaluation effort for similarity ranking and retrieval of scientific workflows. Specifically, we make the following contributions:
1. We introduce a conceptual framework that clearly separates the various tasks of workflow comparison, and use it to re-implement a comprehensive set of existing similarity measures.
2. We present an expert-generated corpus of over 2000 similarity ratings for scientific workflows contributed by 15 scientific workflow experts from four international groups - an effort which, to the best of our knowledge, has not been made public before at this extent.
3. We evaluate several algorithms, both annotational and structural ones, on the collected corpus of similarity ratings, showing how each of their steps contributes to the algorithms' quality, and repeat previous experiments where possible. We also investigate how different similarity measures can be successfully combined in ensembles.
4. We additionally investigate how knowledge derived from the repository as a whole can be applied to structural workflow comparison, and show that these modifications benefit result quality or reduce computational complexity, or both.

In the following, we first introduce our framework for workflow comparison and, reviewing different published methods, show how they were implemented in this framework. In Section 3 we give an overview of previous findings on assessing workflow similarity measures. Our experimental setup, including creation of the gold standard corpus, is described in Section 4, followed by presentation of our evaluation results in Section 5. We conclude in Section 6.

2. A FRAMEWORK FOR SCIENTIFIC WORKFLOW SIMILARITY

The functionality of a scientific workflow is determined by the data processing modules it is composed of, and how these modules are connected by datalinks. While we are ultimately interested in comparing whole workflows, each module represents a distinct functional entity in its own right. As all previous work, we make the reasonable assumption that modules are deterministic and two identical instances of a module are functionally equivalent. Yet, any two different modules may carry the same or very similar functionality. For instance, two different web-services, or a web-service and a locally invoked script, may be functionally equivalent. Thus, to compare scientific workflows wrt their functionality, similarity has to be established on two levels: the level of single modules and the level of whole workflows.

Following this dichotomy, the process of workflow comparison can be conceptually divided into a series of interdependent subtasks: First, the similarity of each pair of modules from two workflows has to be determined by *pairwise module comparison*. Second, using these pairwise module similarities, a *mapping of modules* onto each other needs to be established. Third, the established mapping is used for *topological comparison of the two entire workflows*. Finally, *normalization* of the derived similarity value wrt the sizes of the compared workflows may be desirable.

Note that this setup is not unrelated to that of relational schema matching [31], where mappings between attributes and relations are established. Yet, the elements

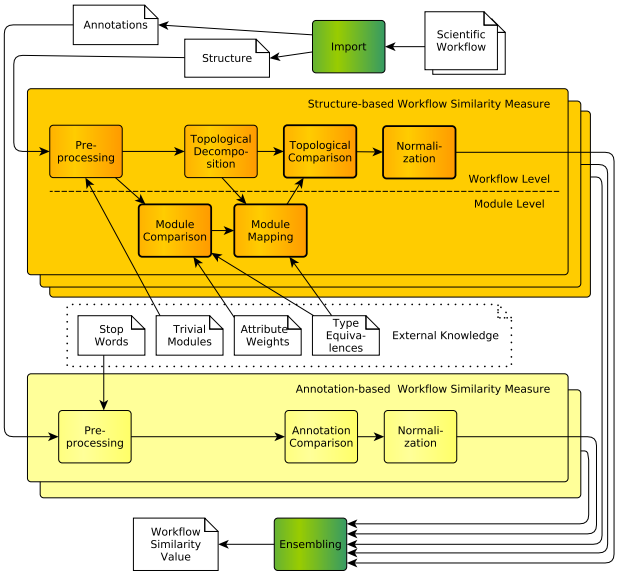


Figure 2: Scientific workflow similarity framework

compared and the underlying intention are quite different: While schema matching compares attributes and relations to establish (mostly) one-to-one equivalences, we here use mappings between modules based on their attributes to derive similarity of entire workflows. Workflow modules are also much richer objects compared to attributes in a schema. Finally, in contrast to schemas workflows have a direction (from source to sink) that is functionally important and that could be exploited for similarity assessment.

Figure 2 drafts the comparison process in the context of our similarity framework. This framework implements the identified steps and allows to uniformly apply and combine them. When two workflows are submitted to the framework, their structure and annotations are separated and made available to the corresponding methods of comparison. Before the actual comparison is done, preprocessing of the underlying data may be applied. For annotations, such preprocessing includes the removal of stop words from the workflow descriptions. Using external knowledge derived from the repository, the workflows' structure can be preprocessed in a similar manner. We will explore the application of such repository-derived knowledge more closely in Section 2.1.5. As for the structural comparison process, we refine the task of *topological comparison* by preceding it by a step of *topological decomposition* of the workflows suitable for the intended comparison. This is useful, for instance, when topological comparison is based on substructures of workflows, e.g., subgraphs or paths. The workflows will then first be decomposed into the respective substructures, on which the *module mapping* is performed. The framework is completed by the option of using either a single similarity measure or an ensemble of two or more measures to derive the overall workflow similarity.

In the following, we look at each of the proposed steps of workflow comparison and how they are approached in previous work more closely. An overview on previous work reflecting the diversity of approaches taken so far can be found in Table 1. We restrict our study to methods targeting the modules, structure and/or annotations of workflows as this is the type of data that can be found in current repositories. While approaches making use of workflow provenance or rich semantic annotations on workflows and modules have been studied in related areas, e.g., [3, 5, 7], workflows currently found in public repositories are typically not associated with such data.

2.1 Structure-based Measures

2.1.1 *Pairwise Module Comparison*

In order to apply any of the structural methods for comparing whole workflows, a way of comparing the workflows' elements is needed. Previous work largely relied on module identification by matching of labels [33, 18, 38], i.e., the names given to a specific instance of a module by a workflows author. In workflow repositories with a heterogeneous author base, however, modules are bound to be labeled non-uniformly. One way of dealing with this heterogeneity is to resort to the matching of other, less varying attributes, such as the modules' types [17]. Another option is to compare labels by their Levenshtein edit distance [4]. To take full advantage of the information contained in a modules specification, however, it seems advantageous to compute module similarity based on a variety of attributes associated with each module, as done in [34]. Which attributes are present in a given module largely depends on the type of operation it provides. For instance, the uri of a web-service to be invoked will only be present in modules designed to invoke a web-service but not in modules performing local operations. Thus, for maximum flexibility, both the set of attributes to compare and the methods to compare them by are configurable in our framework, together with the weight each attribute has in computation of overall module similarity. This approach subsumes all previously proposed methods for module comparison by using appropriate configurations. In our evaluation (see Section 5), we shall investigate the following configurations *pX* for module comparison:

pw0, used as a default, assigns uniform weights to all attributes, and compares module *type*, and the web-service related properties *authority name*, *service name*, and *service uri* using exact string matching. Module *labels*, *descriptions*, and *scripts* in scripted modules are compared using Levenshtein edit distance [23].

pw3 compares single attributes in the same way as *pw0* but uses higher, but not uniform weights for *labels*, *script* and *service uri*, followed by *service name* and *service authority* in the order listed, similar to [34].

pll disregards all attributes but the *labels*, and compares them using the Levenshtein edit distance, resembling the approach taken in [4].

plm disregards all attributes but the *labels*, and compares them using strict string matching as done in [33, 18, 38].

2.1.2 *Module Mapping*

After generating all pairwise module similarities, a *mapping* of the modules has to be established. Such a mapping selects the best similarity match between the modules from the two compared workflows. When using strict matching of singular module attributes to derive module similarity [33, 18, 38, 17], such as the labels, the module mapping is implicitly given as the set of matching modules. When module

Table 1: Existing approaches to scientific workflow comparison and their treatment of comparison tasks.

Ref.	Annotation-based	Structure-based			
		Module Comparison	Module Mapping	Topological Comparison	Normalization
[11]	bag of words				
[36]	frequent tag sets				
		singular attributes	-	frequent module sets	-
[34]		multiple attributes	greedy	sets of modules	$\|V\|$ of smaller workflow
[4]		semantic annotations	maximum weight	sets of modules and edges	$\|V\| + \|E\|$ of query workflow
		label edit distance	maximum weight	sets of modules and edges	$\|V\| + \|E\|$ of query workflow
[33]		label matching	-	vectors of modules	-
		label matching	-	MCS	$\|V\| + \|E\|$ of larger workflow
[18]		label matching	-	MCS	-
		label matching	-	MCS	workflow sizes
[17]		type matching	-	sets of modules	-
		type matching	-	MCS	-
		type matching	-	graph kernels	-
[38]		label matching	-	GED	-

comparison is based on more complex similarity assessment, the best similarity match between the modules of the two compared workflows has to be found explicitly. Previous approaches to this task include greedy selection of mapped modules [34] and computation of the mapping of maximum overall weight [4], both of which have been included in our framework. For clarity of presentation, in the following we only refer to the latter approach of *maximum weight matching (mw)*. We compare both approaches in Section 5.1.3.

Additionally, when an order of the modules to be mapped is given by the *topological decomposition* of the workflows, their *maximum weight non-crossing matching (mwnc)* [27] can be determined to take the given order of modules into account. That is, given two module orderings $(m_1, ..m_i, ..m_k)$ and $(m'_1, ..m'_j, ..m'_l)$, a mapping of maximum weight is computed where the result cannot contain two mappings (m_i, m'_j) and (m_{i+x}, m'_{j-y}) with $x, y \geq 1$.

2.1.3 Topological Workflow Comparison

Regarding topological comparison of scientific workflows, existing approaches can be classified as either a) structure agnostic, i.e., based only on the sets of modules present in two workflows, [34, 33, 36, 4]; b) based on substructures of workflows, such as maximum common subgraphs [33, 18, 17] or graph kernels derived from frequent subgraphs [17]; or c) using the full structure of the compared workflows [38]. We include an approach to topological comparison for each of these classes. We denote the DAG of a workflow as $G_{wf} = (V_{wf}, E_{wf})$.

Sets of Modules – Analogous to the similarity measure described in [34, 33, 36, 4], two workflows $wf1$ and $wf2$ are treated as sets of modules. The additive similarity score of the module pairs mapped by *maximum weight matching (mw)* is used as the non-normalized workflow similarity $nnsim_{MS}$, with $sim(m, m')$ denoting a module pair's similarity value:

$$nnsim_{MS} = \sum sim(m, m') \mid (m, m') \in mw(V_{wf1}, V_{wf2})$$

Sets of Paths – As a slightly relaxed version of using the maximum isomorphic subgraph for workflow comparison [33, 18, 17], the sets of all paths two DAGs are comprised of can be used to compare them by their maximum similar subgraph [22]. We follow this notion and *topologically decompose* each workflow into its set of paths: Starting from each node without inbound datalinks (the DAGs source nodes), all possible paths ending in a node without further outbound links (the DAGs sink nodes) are computed. All pairs (P, P') from the so collected sets of paths PS_{wf1} and PS_{wf2} are compared using the *maximum weight non crossing matching* scheme ($mwnc$) to determine the additive similarity score for each pair of paths:

$$sim(P, P') = \sum sim(m, m') \mid (m, m') \in mwnc(V_{wf1}, V_{wf2})$$

To determine the maximum non-normalized similarity of the two workflows wrt their so compared sets of paths, a *maximum weight matching (mw)* of the paths is computed on the basis of these pairwise path similarity scores:

$$nnsim_{PS} = \sum sim(P, P') \mid (P, P') \in mw(PS_{wf1}, PS_{wf2})$$

Graph Edit Distance – Analogous to the work presented in [38], the full DAG structures of two workflows are compared by computing the graph edit distance using the SUBDUE [29] package. SUBDUE allows labels to identify nodes in a graph, which it uses during the graph matching process. To transform similarity of modules to identifiers, we set the labels of nodes in the two graphs to be compared to reflect the module mapping derived from *maximum weight matching* of the modules during conversion of the workflows to SUBDUE's input format.

The workflows' non-normalized similarity is then computed as

$$nnsim_{GED} = -cost_{GED}$$

For computing graph edit distance, we keep SUBDUE's default configuration which defines equal costs of 1 for any of the possible edit operations (as in [32]). Testing several different weighting schemes did not produce significantly different results.

2.1.4 Normalization

Whether or not to normalize the similarity values derived from topological workflow comparison and how it is to be done strongly depends on the intended use case. When, as in our study, the interest is to determine overall workflow similarity, the goal of the normalization step will be to maximize the information about how well two workflows match globally. As an example, consider two sets of two workflows each, containing 2 and 3 modules, and 98 and 99 modules,

respectively, compared by graph edit distance. If in both cases the workflows match perfectly, with the exception of 1 module and 1 edge, the graph edit distance will be 2 in both cases. Yet, intuitively, the similarity of the workflows in the second set would be deemed higher. Indeed, experimental evaluation showed that normalization wrt workflow size provides significantly better results (see Section 5.1.3).

The step of normalization has been approached rather heterogeneously in previous work (see Table 1). Next to the consideration of workflow size taken therein, our general intention is to acquire similarity values in the range of [0,1]. For set based topological comparisons we resort to a variation of the Jaccard index for set similarity: The Jaccard index is used to express the similarity of two sets A and B by their relative overlap $\frac{|A \cap B|}{|A \cup B|}$ which is equivalent to $\frac{|A \cap B|}{|A|+|B|-|A \cap B|}$. Our modification reflects the fact that mapped elements of our sets of modules or paths are mapped by similarity, not identity. Thus, the size $|V_{wf1} \cap V_{wf2}|$ of the overlap between the two module sets, for instance, is replaced by the overlaps maximum similarity score captured by $nnsim_{MS}$ to derive the overall module set similarity of two workflows:

$$sim_{MS} = \frac{nnsim_{MS}}{|V_{wf1}| + |V_{wf2}| - nnsim_{MS}}$$

The rational here is that where the classical Jaccard index compares the number of mutual elements in the sets with their overall number of (distinct) elements, our modification compares the amount of similarity of similar elements from the sets with the non-similar remainder. If two workflows are identical, each module has a mapping with a similarity value of 1. Then $nnsim_{MS} = |mw(V_{wf1}, V_{wf2})| = |V_{wf1}| = |V_{wf2}|$, and $sim_{MS} = 1$.

For path sets, the normalization is analogous with $|PS_{wf}|$ and $nnsim_{PS}$.

For graph edit distance, we normalize the obtained edit cost by the maximum cost possible. This maximum cost depends on the costs assigned to each edit operation. For our configuration of uniform costs of 1, we thus use the following normalization:

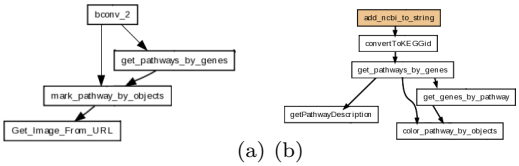
$$sim_{GED} = 1 - \frac{cost_{GED}}{max(|V_{wf1}|, |V_{wf2}|) + |E_{wf1}| + |E_{wf2}|}.$$

The rational here is that in the worst case, each node in the bigger set of nodes is either substituted or deleted, while for the edges a complete set of insertions and deletions may be necessary.

2.1.5 Including Repository Knowledge

Knowledge derived from a repository of scientific workflows [37, 35], can be used in the structural comparison process. We investigate two possible applications of such knowledge.

Module Pair Preselection – When comparing sets of modules V_{wf1} and V_{wf2} from two scientific workflows, the general approach is to compare each pair of modules from the Cartesian product $V_{wf1} \times V_{wf2}$ of the workflows' module sets. To reduce the amount of module pair comparisons, restrictions can be imposed on candidate pairs by requiring certain module attributes to match in a certain sense. As modules of the same *type* are more likely to carry similar functionality, a first strategy for candidate selection requires module *types* to match. As a second, less strict selection strategy, we cast module *types* to equivalence classes based on the categorization proposed in [37]. For instance, one such class holds all web-service related module types. Modules within each such class may be compared and mapped onto each other. This introduction of *type equivalences* was motivated by the observation that in the used dataset of Taverna workflows, especially web-service modules are typed with a variety of identifiers, such as 'arbitrarywsdl', 'wsdl', or 'soaplabwsdl'.



(a) (b)

Figure 3: Sample importance projection of scientific workflow 1189 (a) and 2805 (b) (see also Fig. 1).

Importance Projection Preprocessing – Not all modules in a scientific workflow contribute equally to the workflows specific functionality: Modules used most frequently across different workflows often provide trivial, rather unspecific functionality, such as splitting a string into a list of strings [35]. To account for this, we assign a score to each module indicating the importance of the module for a workflow's specific functionality. Only modules with a score above a configurable threshold are kept for further use in module and workflow similarity computation. The workflow is thus projected onto its most relevant modules. In order to make full use of this projection in all our structural similarity measures, all paths between important modules are preserved as edges in terms of the transitive reduction of the resulting DAG. That is, if two important modules are connected by one or more paths along non-important modules in the original workflow, they are connected by one edge in the *Importance Projection (ip)*. Figure 3 shows the resulting projection of our example workflows 1189 and 2805. The selection of important modules is currently done manually based on module types. Modules that perform predefined, trivial local operations are removed.

2.2 Annotation-based Measures

Purely annotation-based methods use only textual information recorded with the workflows in a repository. Such information includes the workflow's title, a free form text description, and assigned keyword tags. Two approaches to annotational similarity of scientific workflows have been proposed which we include in our framework:

Bag of Words – Following the work of [11], workflows are compared by their titles and descriptions using a bag-of-words approach. Both title and description are tokenized using whitespace and underscores as separators. The resulting tokens are converted to lowercase and cleansed from any non alphanumeric characters. Tokens are filtered for stopwords. The workflows' similarity is then computed as

$$sim_{BW} = \frac{\#matches}{\#matches + \#mismatches}$$

where *#matches* is the number of tokens found in both workflows' title or description, and *#mismatches* is the number of tokens present only in either one workflow. This quotient again corresponds to the Jaccard index for set similarity. Please note that our algorithm presented here does not account for multiple occurrences of single tokens. We did try variants that do so, but evaluation showed that these variants performed slightly worse than the one described here (data not shown).

Bag of Tags – The keyword tags assigned to scientific workflows in a repository can also be used for similarity assessment, as done in [36]. The tags assigned to a workflow are treated as a bag of tags and calculate workflow similarity in the same way as in the *Bag of Words* approach described above. Following the approach of [36], no stopword removal or other preprocessing of the tags is performed. This also aligns with our expectation of tags to be specifically preselected by the workflow author.

3. PREVIOUS FINDINGS

After the in depth review of existing approaches to the various steps of scientific workflow comparison in the previous section, we here summarize results of all previous evaluations we are aware of. The concrete type of evaluation the proposed methods were subjected to varies with the intended use case, e.g., clustering or similarity search. Additionally, evaluation settings vary greatly from manual inspection of a methods output [11, 36, 34, 18] to systematic evaluation against a small (and not publicly available) set of expert provided truth [4] or against ground truth derived from another method [17]. The following findings have been reported:

Module Comparison – [4] found that similarity based on semantic annotations of workflow modules provides better results in workflow retrieval than using module similarity based on edit distance of their labels. Note that in this work, semantic annotations were manually assigned to modules, whereas scientific workflows in public repositories usually don't contain such semantic information. Comparing modules by matching of labels, [18] found the lowercasing of labels to slightly improve ranked retrieval on the example of a single query workflow.

Topological Comparison – [34] found workflows regarded similar by comparison of their sets of modules to also be structurally similar upon manual inspection, indicating a correlation between the modules used in a workflow and its topology. Similarly, comparing the use of maximum common isomorphic subgraphs (MCS) and module label vectors for workflow comparison, [33] found that both methods deliver similar results. Conversely, [17] found MCS to perform better than bags of modules, and both of them to be slightly outperformed by graph kernels.

Normalization – [18], as the only study we are aware of to provide comparative information about this aspect, suggests that normalization wrt workflow size improves results of scientific workflow comparison.

Annotational Comparison – Approaches based on workflow annotations have been generally reported to deliver satisfying results [11, 36, 17]. They have been found to perform either as good as or better than structural approaches in workflow comparison [36, 17].

4. EXPERIMENTAL SETUP

The goal of this work is an unbiased and comprehensive benchmark of different methods for similarity search in a corpus of scientific workflows. To this end, we reimplemented all methods described in Section 3, structured according to the steps of our comparison framework. We collected a quite comprehensive corpus of similarity ratings for a set of Taverna workflows from the myExperiment scientific workflow repository [32]. A second set of workflows was assembled from the Galaxy online repository [20]. Gold standard ratings were obtained from 15 field experts and aggregated using median ranking. We compare all algorithms regarding *ranking correctness* and *retrieval precision*. These topics are discussed in more detail in the following. Note that all data will be made publicly available.

4.1 Workflow Dataset

myExperiment provides a download of all workflows publicly available in its repository. For Taverna workflows, which make up approx. 85% of myExperiment [35], this dataset contains an rdf representation of the workflow structure, along with attributes of the modules used and annotations provided by authors. We transformed all workflows into a custom graph format for easier handling. During this transformation, subworkflows were inlined and input and output ports were removed. The complete dataset contains 1483 workflows. myExperiment - and thus our dataset - is generally domain agnostic. To match the expertise of our expert curators, we focussed on workflows from the life sciences domain in the evaluation. We will discuss possible implications of this decision in Section 6.

To investigate transferability of findings to other workflow formats, we also created a secondary eval data set of 139 Galaxy workflows from the public Galaxy repository. These workflows were transformed into the same internal format as described above and processed using the exact same methods. We performed most experiments with the larger myExperiment data and used the Galaxy data set as independent validation and to study data set specific properties and their implications on algorithm performance (see Section 5.3).

4.2 Expert Ratings

We conducted a user study to collect manually assigned similarity ratings for selected pairs of scientific workflows. Overall, 15 domain experts from six different institutions participated. Ratings were obtained in two phases:

In a first experiment, the goal was to generate a corpus of ratings independent of a concrete similarity measure to make it suitable for evaluation of large numbers of different measures, and measures to be developed in the future. 24 life science workflows, randomly selected from our dataset, (called *query workflows* in the following) were presented to the users, each accompanied by a list of 10 other workflows to compare it to. To obtain these 10 workflows, we ranked all workflows in the repository wrt a given query workflow using a naive annotation based similarity measure and drew workflows at random from the top-10, the middle, and the lower 30. The ratings were to be given along a four step Likert scale [24] with the options *very similar*, *similar*, *related*, and *dissimilar* plus an additional option *unsure*. *Unsure*

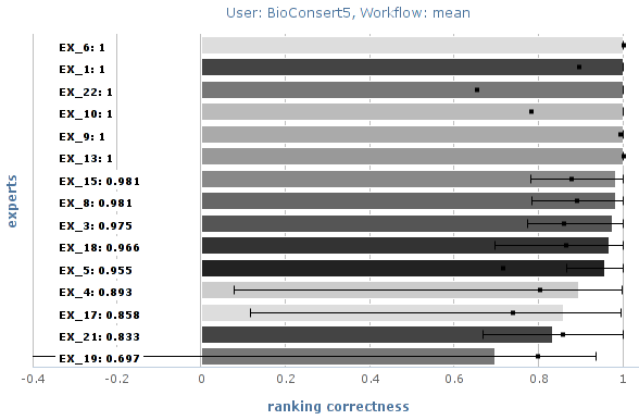


Figure 4: Mean ranking correctness (bars) with upper and lower stddev (errorbars), and mean ranking completeness (black squares) for single experts' rankings compared to the ranking derived as BioConsert expert consensus.

user ratings were not further considered in the evaluation. The ratings collected in this first experiment were used to rank the 10 workflows for each of the query workflows. The individual experts' rankings were aggregated into consensus rankings using the BioConsert algorithm [9], extended to allow incomplete rankings with *unsure* ratings. On the basis of the generated consensus rankings, we evaluate the algorithms' *ranking correctness* in Section 5.1. Figure 4 inspects inter-annotator agreement, comparing each single expert's rankings to the generated consensus using the ranking correctness and completeness measures described below in Section 4.3. While we do see a few outliers, most experts are rather d'accord about how workflows are ranked.

In a second experiment, the selected algorithms were run to each retrieve the top-10 similar workflows from our complete dataset of 1483 Taverna workflows for eight of the 24 query workflows from the first experiment. The results returned by each tested algorithm were merged into single lists between 21 and 68 elements long (depending on the overlap in the algorithm's top-10). Naturally, these lists did contain workflows already rated in the first experiment. Experts were now asked to complete the ratings using the same scale as before. The ratings provided in this second experiment qualify each workflow in the search results of each of the used algorithms wrt their user-perceived similarity. Using these completed ratings we evaluate the algorithms' *retrieval precision* in Section 5.2 (definition below). Different experts' opinions were aggregated as the median rating for each pair of query and result workflow.

Altogether, we presented 485 workflow pairs (24 x 10 from the first experiment and 255 non-overlapping pairs from the second) to the 15 experts and received a total of 2424 ratings.

4.3 Evaluation Metrics

As proposed in previous work [4], we use the measures of ranking *correctness* and *completeness* [8] to compare the algorithms' rankings against the experts' consensus rankings in our first experiment. To evaluate the algorithms' retrieval performance on the ratings from our second experiment, we compute *precision at k* of the algorithmic search results.

Ranking Correctness – For ranking correctness, the order of each pair of elements in the rankings is compared. If in both rankings the elements are not tied and their order is the same, the pair's order is called *concordant*. If their orders differ, the pair is *discordant*. Pairs tied in either of the two rankings do not count for correctness, which is computed as

$$correctness = \frac{\#concordant - \#discordant}{\#concordant + \#discordant}$$

where $\#concordant$ and $\#discordant$ are the numbers of concordant and discordant pairs, respectively. Correctness values range from -1 to 1, where 1 indicates full correlation of the rankings, 0 indicates that there is no correlation, and negative values are given to negatively correlated rankings.

Ranking Completeness – Ranking completeness, on the other hand, measures the number of pairs of ranked elements that are not tied in the expert ranking, but tied in the evaluated algorithmic ranking.

$$completeness = \frac{\#concordant + \#discordant}{\#pairs\_ranked\_by\_experts}$$

The objective here is to penalize the tying of elements by the algorithm when the user distinguishes their ranking order.

Retrieval Precision at k – The algorithms' retrieval precision at k is calculated as

$$P@k(result\ list) = \frac{1}{k}\sum_{i=1}^{k} rel(r_i)$$

at each rank position $0 < k \leq 10$, with $rel(r_i) \in 0, 1$ being the relevance of the result at rank i. As the expert ratings on whether a workflow is a relevant result for similarity search with a given query workflow are quaternary by the Likert scale used, we consider different relevance thresholds: *very similar*, *similar* or *related*. For instance, only workflows with a median rating of *similar* are regarded to be relevant.

5. RESULTS

We now present the results obtained from the two experiments on algorithmic *workflow ranking* and *workflow retrieval*. We start with a baseline evaluation, focussing on the methods of comparing workflows proposed in previous work: *Sets of Modules* [34, 33, 36], workflow substructures [33, 18, 17] in terms of their *Sets of Paths* [22], *Graph Edit Distance* [38], *Bags of Words* [11], and tag-based workflow comparison [36, 17] as *Bags of Tags*. We then investigate each step of workflow comparison in detail, from different approaches to module comparison to the inclusion of external knowledge in the comparison process. As we will see, each of these steps plays a defining role for result quality, and using the right settings can significantly improve result quality.

5.1 Workflow Ranking

5.1.1 Baseline Evaluation

Figure 5 shows mean ranking correctness and completeness over all query workflows for each of the similarity algorithms under investigation. Note that for *Graph Edit*

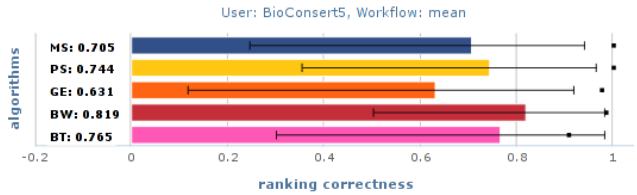


Figure 5: Mean ranking correctness (bars) with upper and lower stddev (errorbars), and mean ranking completeness (black squares) for similarity algorithms against BioConsert expert consensus. Numerical values denote mean average correctness.

Table 2: Algorithm shorthand notation overview

Notation	Description
MS	*Module Sets* topological comparison
PS	*Path Sets* topological comparison
GE	*Graph Edit Distance* topological comparison
BW	*Bag of Words* annotation based comparison
BT	*Bag of Tags* annotation based comparison
np	No structural preprocessing of workflows
ip	*Importance projection* workflow preprocessing
ta	No module pair preselection for comparison
te	*Type equivalence* based module pair preselection
pw0	Module comparison with uniform attribute weights
pw3	Module comparison on tuned attribute weights
pll	Module comparison by edit distance of labels only
plm	Module comparison by matching of labels only

Distance, we allowed match cost computation of each of the 240 pairs of scientific workflows to take a maximum of 5 minutes. 23 pairs not computable in this timeframe were disregarded in the evaluation. All algorithms are used in their basic, normalized configurations with uniform weights on all module attributes. The sim_{BW} algorithm has the best results in terms of ranking correctness. sim_{BT} and the structural similarity measure sim_{PS} almost tie, followed by sim_{MS}. sim_{GE} delivers worst performance and is the only algorithm in this set with a statistically significant ($p<0.05$, paired ttest) difference to sim_{BW}. These findings largely confirm those of previous work (see Section 3) that annotational measures outperform structural workflow similarity — in certain settings. The good performance of sim_{BW} is not surprising: Well written titles and descriptions capture the main functional aspects of a workflow, and provide a strong means for judging workflow similarity.

Interestingly, while most structural similarity measures are complete in their ranking, both annotational measures tie some of the compared workflows where experts see them differently ranked. This is especially true for sim_{BT}, which, additionally, is not able to provide rankings for four of the given query workflows due to lack of tags. These workflows were not considered for computation of sim_{BT} ranking performance. Note that around 15% of the workflows in our complete dataset lack tags.

As for the different structural comparison methods, sim_{GE} is clearly outperformed by the other two. This indicates that many functionally similar workflows, while sharing modules and substructures, do differ in their overall graph layout. We will see this observation confirmed in Section 5.1.4, when workflows are preprocessed to remove structural noise.

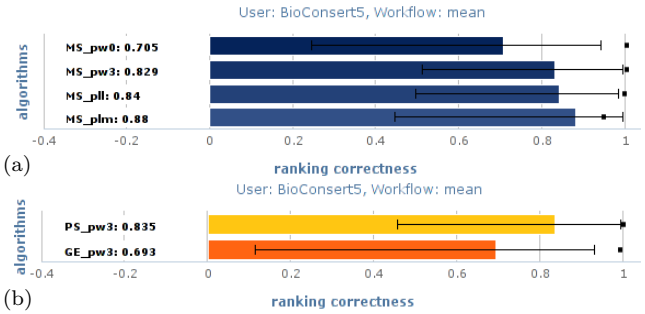


Figure 6: Mean ranking correctness for (a) sim_{MS} with various module attribute weightings, and (b) sim_{PS} and sim_{GE} with $pw3$.

With these initial evaluation results as a baseline, we inspect several aspects of workflow comparison for their impact on ranking performance.

5.1.2 Module Comparison (pX)

Figure 6a shows the impact of the different module similarity schemes (see Section 2.1.1 and Table 2 for an overview of the used notation) on ranking correctness by the example of the sim_{MS} algorithm. Trends are similar for the other measures (data not shown). Clearly, the uniform weighting scheme *pw0* used in the baseline evaluation of Figure 5 performs worst ($p<0.05$). Using only the edit distance of *labels* for module comparison (*pll*) is on par with the more complex scheme *pw3* (using refined weights on various attributes) in terms of mean ranking correctness. Note that there is a minimal reduction in ranking completeness for *pll*, resulting from the less fine grained similarity assessment. This reduction in completeness is much more pronounced when using label matching (*plm*) for assessing module similarity. Further investigation showed that the striking increase in ranking correctness *plm* achieves is in fact due to this reduction in ranking completeness, as pairs of workflows tied by the algorithmic ranking are not accounted for when determining ranking correctness: while the most similar workflows are ranked high, less similar ones are not distinguished in terms of their similarity any more.
The (to us) surprisingly good performance of the *label* only approaches, especially *pll*, shows that while the author base of the workflows in our dataset is heterogeneous, the labels given to modules mostly do reflect their function and thus convey measurable information about their similarity. Yet, the strict matching of labels, as used in many previous studies, offers to little fine grained similarity for differentiated assessment of workflow similarity. We will see how this observation affects workflow retrieval in Section 5.2.

The change in the algorithms' ranking performance induced by both the *pw3* and *pll* weighting schemes (see Fig. 6a and b) puts these algorithms ahead of sim_{BW}, if yet not significantly. The exception here is sim_{GE}, where the impact of different weighting schemes is less pronounced. The reason for this most probably lies in the specific type of structural comparison applied: The more overall workflow topology is taken into account, the less do differences in similarity assessment of single pairs of modules matter.

From here on, we denote the module weighting scheme

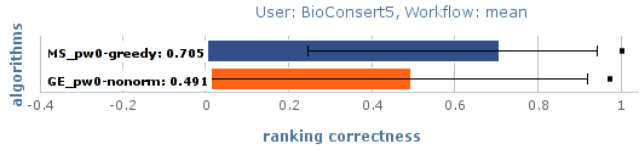


Figure 7: Mean ranking correctness for sim_{MS} with greedy mapping of modules and sim_{GE} without normalization of edit distance.

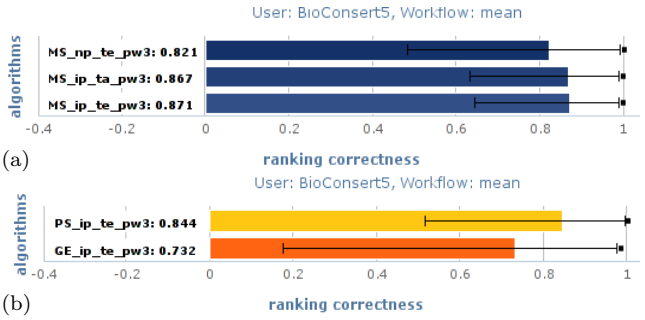


Figure 8: Mean ranking correctness for (a) sim_{MS}, and (b) sim_{PS} and sim_{GE} when including external knowledge.

used with an algorithm by specifying the corresponding pX in its name (e.g., *GE_pw3* refers to the sim_{GE} using the $pw3$ module attribute weighting scheme for module comparison).

5.1.3 Module Mapping and Normalization

We investigated the impact of different module mapping strategies and of using normalization or not. Figure 7 exemplifies our findings on two settings that have been used in previous approaches: (1) Greedy mapping of modules in the *Module Sets* measure [34] has no impact on ranking quality when compared to using maximum weight matching (see Fig. 5). Remarkably, this indicates that in the set of workflows studied, the mappings of modules are rather non-ambiguous, i.e. modules mostly have a single best mapping partner in the respective other workflow. (2) The omission of normalization of similarity values with *Graph Edit Distance* [38], on the other hand, significantly reduces ranking correctness ($p<0.05$) when compared to results for normalized sim_{GE} (see Fig. 5). This data confirms the results from [18], but on a much larger data base.

5.1.4 Including Repository Knowledge

Module Pair Preselection (tX) – Figure 8a (first bar) shows the impact of the *type equivalence (te)* module pair preselection strategy proposed in Section 2.1.5 for sim_{MS}. The trends of these changes are similar for all algorithms: While strict type matching significantly decreases the algorithms' ranking correctness (data not shown), the use of equivalence classes results in correctness values comparable to those without any restrictions on module pair selection. This is remarkable, as it shows that a) the technical classes of modules (*web-service, script, local operation*) do play an important role in determining a modules function; and b) this technical distinction of modules is also detected when comparing all pairs of modules - even when using only labels.

While *te* doesn't lead to an improvement of user perceivable ranking quality, the exclusion of certain module pairs

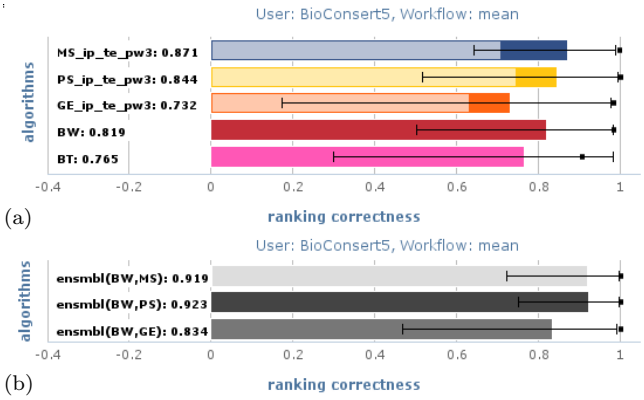


Figure 9: Mean ranking correctness for (a) single algorithms' best configurations (shaded: baseline evaluation, see Fig. 5), and (b) the best ensembles of two (see text).

from comparison yields a reduction of such pairwise module comparisons by a factor of 2.3 (172k/74k on the evaluation dataset of experiment 1) and thus a notable runtime reduction. From here on, we denote the module pair selection used with an algorithm by specifying '*te*' for equivalence class based selection and '*ta*' for selection of all pairs of modules in the algorithm's name.

Importance Projection (Xp) – As shown in Figure 8a and b, most algorithms benefit from adding *Importance Projection (ip)* preprocessing (Section 2.1.5), with the exception of sim_{PS} showing stable results. The reduction in structural noise is especially visible in sim_{GE}: As different ways of transforming intermediate results within the workflow are removed, similarities in the constellations of the most specific functional modules become more pronounced. This positive impact of *ip* on the correlation of algorithm rankings with expert rankings confirms our intention of removing presumably unimportant modules, and confirms our selection of modules to keep. Yet, such manual selection is only possible with both in depth understanding of the types of modules used in a dataset of scientific workflows, and knowledge about their specific relevance to a workflow's functionality. An interesting line of research to be explored in future work are methods to perform such a preselection automatically, for instance, based on module usage frequencies.

As a side effect, *ip* leads to a decrease in the average number of modules per workflow from 11.3 to 4.7 in our dataset, resulting in a significant increase in computational performance of all structural algorithms. This effect is especially relevant for *Graph Edit Distance* computation: where sim_{GE} was able to compute the similarities of only 217 of the 240 workflow pairs of our ranking experiment within the per-pair timeout of 5 minutes without using *ip*, it can now compute all pairs but one. From here on, we denote the use of *importance projection* with an algorithm by specifying '*ip*' in its name, and '*np*' for its omission.

5.1.5 Best Configurations

The three modifications *pX (module comparison scheme)*, *tX (module pair preselection)*, and *Xp (importance projection preprocessing)* can be used in all combinations with

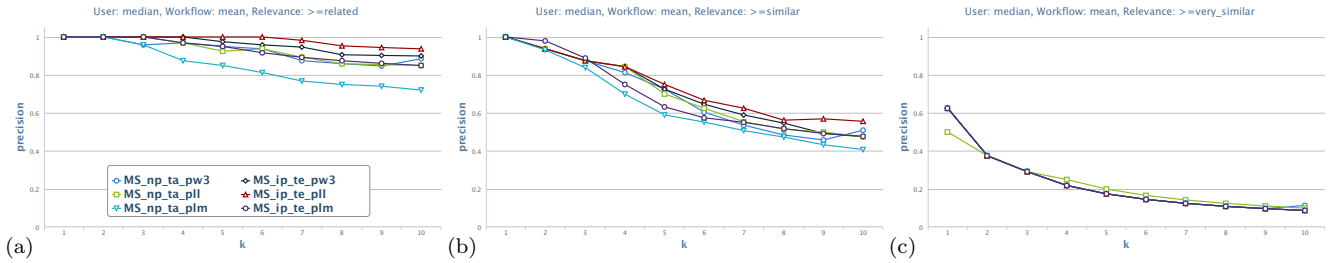


Figure 10: Mean retrieval precision at k against the median expert rating for sim_{MS} in various configurations of module similarity assessment (pX), with and without *ip* and *te* for relevance threshold (a) related, (b) similar, and (c) very similar.

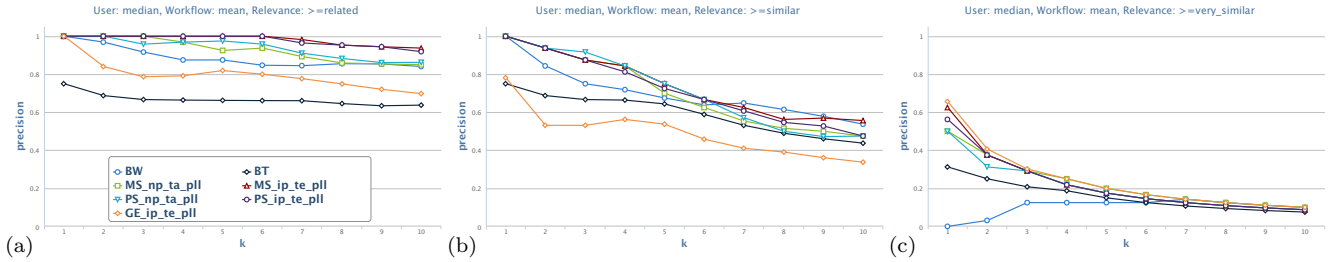


Figure 11: Mean retrieval precision at k of structural and annotational similarity algorithms for median expert rated relevance of (a) related, (b) similar, and (c) very similar.

each of the algorithms, resulting in a total of 72 different configurations (not considering different methods for module mapping and normalization). For each algorithm, Figure 9a shows the configuration with its best standalone ranking performance in direct comparison to the annotation-based approaches: When tuned appropriately, algorithms based on workflow structure outperform annotation based approaches, except when very strict graph comparison is applied as in sim_{GE}. Note that the differences between *pw3* and *pll* are minimal and not significant.

5.1.6 Ensembles of Algorithms

Just as consensus can be generated by aggregating expert rankings, the rankings produced by the similarity algorithms can be combined into a single ranking. We tested such ensembles by simply taking the average of the scores of selected individual ranking algorithms. This especially allows to integrate the different perspectives of annotational and structural workflow similarity. We ran experiments for all combinations of two algorithms and indeed found the best performing ensembles to aggregate sim_{BW} and either sim_{MS} or sim_{PS} with *ip*, *te* and *pll*. The resulting improvement of ranking performance of these ensembles over any algorithm used on its own is both significant ($p < 0.05$) and substantial (see Fig. 9b). As implied by the reduction of standard deviation, results are also much more stable.

5.2 Workflow Retrieval

We investigate the algorithms' retrieval performance over a whole repository in terms of *retrieval precision* over the top 10 search results. We found the results of the first experiment mostly confirmed and only report the most interesting findings here. We first inspect results for different module similarity schemes, followed by a comparison of results on the level of whole workflows. The use of external knowledge is taken into account on both levels.

5.2.1 Module Comparison

Figure 10 graphs retrieval precision at k for sim_{MS} with various module similarity schemes pX, with and without *ip* and *te*. Three different relevance thresholds *related*, *similar*, and *very similar* are considered. Interestingly, the differences in retrieval quality decrease with the increase in relevance level up to the point where all configurations deliver similar performance for retrieval of *very similar* workflows. Apparently, finding the most similar results is independent of the module similarity scheme used. For the retrieval of *related* workflows, on the other hand, where more fine grained assessment of similarity is required, differences between the schemes become visible. The strict label matching approach of *plm* performs worst, confirming our observations from the previous experiment. *pw3* and *pll* tie when not using external knowledge. The inclusion of such knowledge improves performance of all configurations, and puts *pll* ahead of *pw3*, both in terms of mean precision and in terms of standard deviation, which is substantially smaller (not shown).

5.2.2 Workflow Similarity

Figure 11 shows retrieval precision for the structural and annotational workflow comparison approaches under evaluation. The structural approaches are used with the *pll* module similarity scheme of edit distance comparison of labels, and have been applied with and without *ip* and *te*. Note that for sim_{GE} we only include results for preprocessed workflow graphs (*ip*). The strict topological comparison applied by sim_{GE} does find the most similar workflows equally well as its structural contenders, but provides much worse results when looking for *similar* or *related* workflows: As sim_{GE} puts a strong emphasis on workflow structure, it also retrieves workflows from other domains than the query workflow, which happen to be similar to the query workflow in terms of their graph structure. sim_{MS} and sim_{PS} provide equivalent result quality and provide best results for both

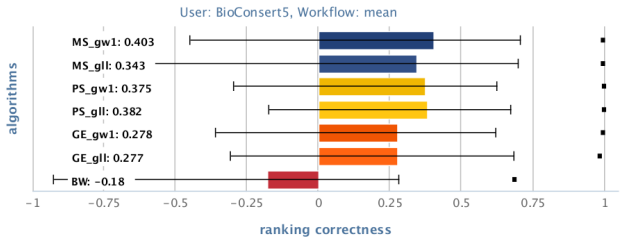


Figure 12: Mean ranking correctness for algorithms with different module comparison schemes (see text) on Galaxy workflows.

related and *similar* workflows. Notice that the difference between configurations with *ip* and *te* and those without is most pronounced for retrieval of *related* workflows, where the inclusion of external knowledge improves both mean precision and stability of the algorithms' performance in terms of standard deviation from the mean. sim_{BW}, while less precise in retrieval than sim_{MS} and sim_{PS}, is best at retrieving *related* and *similar* workflows, while it fails to deliver workflows with a median expert rating of *very similar* within the very top of its search results for the set of workflows studied.

5.3 Evaluation on Second Data Set

Figure 12 shows ranking results on our second dataset of Galaxy workflows, for which we repeated our first experiment on workflow ranking using 8 query workflows. The module comparison schemes used are $gw1$, comparing a selection of attributes with uniform weights, and gll, comparing only module labels by their edit distance. The most striking observation is that sim_{BW} doesn't provide satisfying results on this data set, as the Galaxy workflows carry less annotations. While, overall, results for the structural algorithms are less convincing than on Taverna workflows, our observations are generally confirmed: structure agnostic comparison by sim_{MS} and comparison respecting substructures by sim_{PS} outperform the strict comparison of full workflow structure performed by sim_{GE}. Interestingly, here label-only comparison of modules offers less correct results than comparison of multiple attributes. As previously observed, sim_{PS} provides more stable results.

6. CONCLUSION

In this study, we compared a multitude of existing approaches to scientific workflow comparison on the to-date largest human-curated corpus of similarity ratings for scientific workflows. We paid special attention to deconstruct every method into their most important conceptual steps and to align these to a common framework, with the goal to increase comparability and to be able to pinpoint observed differences in result quality to their precise cause. Our evaluation clearly showed that each step in the process of workflow comparison makes an important contribution to overall result quality. While, for practical reasons, our evaluation did focus on workflows from the life sciences domain, the used algorithms are domain agnostic and do not make use of any domain specific knowledge. We do, however, believe that the life sciences are a particularly difficult domain for workflow comparison, due to the large number of different groups developing different tools (and workflows), even for similar tasks, leading to difficult task-matching issues. Our most important findings are:

1. For module comparison, the edit distance of module *labels* seems to be the best approach: It provides best results in retrieval and does not require refinement of multiple attribute weights; and it provides a more fine grained assessment of similarity than label matching, which, in turn, can only be recommended for retrieval of the few most similar results. Of course, these findings are only valid if labels are telling, i.e., are indicative for the functionality of the labeled module. Such workflows include the studied Taverna workflows from the myExperiment repository, but also the majority of workflows found in the SHIWA repository [2].
2. We have shown that structural approaches can outperform annotational ones when configured appropriately. Especially in repositories where workflows are not well annotated by descriptions or tags, such as the Galaxy repository [20] inspected here, or the CrowdLabs repository of VisTrails workflows [28], structural approaches are indispensible. While full structural comparison by *Graph Edit Distance* appears to be to strict - similar to label matching on the module level -, comparing workflows either by substructures such as paths or by the sets of modules they contain provide comparably convincing results. This is good news, as module set comparison is computationally far less complex than comparing substructures. Yet, the fact that *Path Sets* comparison is more stable in its results across different configurations indicates room for further research to include topological information with less computational complexity.
3. Normalization of the similarity values derived from workflow comparison wrt workflow size is, as clearly shown, indispensable for similarity search of scientific workflows.
4. Next to the intrinsic steps of workflow comparison, we have also looked at several options for further tuning. The use of external knowledge, potentially derived from the workflow repository itself, reduces computational complexity and often improves result quality. Yet, manual acquisition of such knowledge as done in this study, requires extra work to be invested prior to comparisons; furthermore, properties derived from a given repository usually are not transferable to other repositories. Finding suitable ways to automatically derive the required knowledge from the repository, is an interesting area of future research.
5. Another line of future work is to further investigate ensembles of different algorithms: we have shown that such ensembles can significantly improve result quality when compared to single algorithms. The approach of using the algorithms' mean similarity presented here provides a starting ground, leaving room for testing advanced methods such as boosting or stacking [26].

Finally, we plan to apply our framework to similarity search in business model repositories.

7. ACKNOWLEDGMENTS

We sincerely thank K. Belhajjame, J. Brandt, M. Bux, L. Childs, D. Garijo, Y. Gil, B. Haldemann, S. Kröger, P. Larmande, B. Ludäscher, P. Missier, C. Toffano, and K. Zimmermann for contributing workflow similarity ratings. This work was partly funded by DFG grant GRK1651, DAAD grant 55988515 and PHC Procope grant. Work partially done in the context of the Institut de Biologie Computationnelle, Montpellier, France.

8. REFERENCES

[1] ProvBench initiative. *https://sites.google.com/site/provbench/*.

[2] SHIWA workflow repository. *http://shiwa-repo.cpc.wmin.ac.uk*.

[3] Z. Bao, S. Cohen-Boulakia, S. B. Davidson, A. Eyal, and S. Khanna. Differencing provenance in scientific workflows. *ICDE*, pages 808–819, 2009.

[4] R. Bergmann and Y. Gil. Similarity assessment and efficient retrieval of semantic workflows. *Information Systems*, 40:115–127, 2012.

[5] P. Bertoli, M. Pistore, and P. Traverso. Automated composition of web services via planning in asynchronous domains. *Artificial Intelligence*, 174(3):316–361, 2010.

[6] S. Bowers and B. Ludäscher. Actor-oriented design of scientific workflows. *Conceptual Modeling*, pages 369–384, 2005.

[7] A. Brogi, S. Corfini, and R. Popescu. Semantics-based composition-oriented discovery of web services. *Transactions on Internet Technology*, 8(4):19, 2008.

[8] W. Cheng, M. Rademaker, B. Baets, and E. Hüllermeier. Predicting partial orders: ranking with abstention. *ECML/PKDD*, pages 215–230, 2010.

[9] S. Cohen-Boulakia, A. Denise, and S. Hamel. Using medians to generate consensus rankings for biological data. *SSDBM*, pages 73–90, 2011.

[10] S. Cohen-Boulakia and U. Leser. Search, adapt, and reuse: The future of scientific workflow management systems. *SIGMOD Record*, 40(2):6–16, 2011.

[11] F. Costa, D. d. Oliveira, E. Ogasawara, A. Lima, and M. Mattoso. Athena: Text mining based discovery of scientific workflows in disperse repositories. *RED*, pages 104–121, 2010.

[12] E. Deelman, D. Gannon, M. Shields, and I. Taylor. Workflows and e-science: An overview of workflow system features and capabilities. *Future Generation Computer Systems*, 25(5):528–540, 2009.

[13] R. Dijkman, M. Dumas, B. V. Dongen, R. Käärik, and J. Mendling. Similarity of business process models: Metrics and evaluation. *Information Systems*, 36(2):498–516, 2010.

[14] R. Dijkman, M. Dumas, and L. García-Bañuelos. Graph matching algorithms for business process model similarity search. *Business Process Management*, pages 48–63, 2009.

[15] M. Dumas, L. García-Bañuelos, and R. Dijkman. Similarity search of business process models. *Data Engineering Bull.*, 32(3):23–28, 2009.

[16] J. Freire, C. Silva, S. Callahan, E. Santos, C. Scheidegger, and H. Vo. Managing rapidly-evolving scientific workflows. *IPAW*, pages 10–18, 2006.

[17] N. Friesen and S. Rüping. Workflow analysis using graph kernels. *SoKD*, 2010.

[18] A. Goderis, P. Li, and C. Goble. Workflow discovery: the problem, a case study from e-science and a graph-based solution. *ICWS*, pages 312–319, 2006.

[19] A. Goderis, U. Sattler, P. Lord, and C. Goble. Seven bottlenecks to workflow reuse and repurposing. *ISWC*, pages 323–337, 2005.

[20] J. Goecks, A. Nekrutenko, and J. Taylor. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biology*, 11(8):R86, 2010.

[21] J.-Y. Jung, J. Bae, and L. Liu. Hierarchical business process clustering. *Services*, 2:613–616, 2008.

[22] J. Krinke. Identifying similar code with program dependence graphs. *WCRE*, pages 301–309, 2001.

[23] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet physics doklady*, 10:707, 1966.

[24] R. Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 1932.

[25] B. Ludäscher, M. Weske, T. McPhillips, and S. Bowers. Scientific workflows: Business as usual? *Business Process Management*, pages 31–47, 2009.

[26] R. Maclin and D. Opitz. Popular ensemble methods: An empirical study. *arXiv:1106.0257 preprint*, 2011.

[27] F. Malucelli, T. Ottmann, and D. Pretolani. Efficient labelling algorithms for the maximum noncrossing matching problem. *Discrete Applied Mathematics*, 47(2):175–179, 1993.

[28] P. Mates, E. Santos, J. Freire, and C. Silva. Crowdlabs: Social analysis and visualization for the sciences. *SSDBM*, pages 555–564, 2011.

[29] B. Messmer and H. Bunke. Efficient subgraph isomorphism detection: a decomposition approach. *Knowledge and Data Engineering*, 12(2):307–323, 2000.

[30] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, R. Greenwood, K. Carver, M. Pocock, A. Wipat, and L. P. Taverna: a tool for the composition and enactment of bioinformatics workflow. *Bioinformatics*, 20(17):3045–3054, 2003.

[31] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*, 10(4):334–350, 2001.

[32] D. Roure, C. Goble, and R. Stevens. The design and realisation of the myexperiment virtual research environment for social sharing of workflows. *Future Generation Computer Systems*, 25(5):561–567, 2009.

[33] E. Santos, L. Lins, J. Ahrens, J. Freire, and C. Silva. A first study on clustering collections of workflow graphs. *IPAW*, pages 160–173, 2008.

[34] V. Silva, F. Chirigati, K. Maia, E. Ogasawara, D. Oliveira, V. Braganholo, L. Murta, and M. Mattoso. Similarity-based workflow clustering. *JCIS*, 2(1):23–35, 2011.

[35] J. Starlinger, S. Cohen-Boulakia, and U. Leser. (Re)Use in Public Scientific Workflow Repositories. *SSDBM*, pages 361–378, 2012.

[36] J. Stoyanovich, B. Taskar, and S. Davidson. Exploring repositories of scientific workflows. *WANDS*, pages 7:1–7:10, 2010.

[37] I. Wassink, P. Vet, K. Wolstencroft, P. Neerincx, M. Roos, H. Rauwerda, and B. T.M. Analysing scientific workflows: Why workflows not only connect web services. *Services*, pages 314–321, 2009.

[38] X. Xiang and G. Madey. Improving the reuse of scientific workflows and their by-products. *ICWS*, pages 792–799, 2007.

[39] U. Yildiz, A. Guabtni, and A. Ngu. Business versus scientific workflows: A comparative study. *Services*, pages 340–343, 2009.